REMARKS

Claims 1-24 are pending in the present patent application. Applicant has amended claims 1-18. Applicant has cancelled claims 19-22. Applicant has added claims 23 and 24. Applicant respectfully requests reconsideration and re-examination of claims 1-18, 23 and 24 in the present patent application and presents the following arguments:

Examiner's Rejection of Claims 1-18 under 35 U.S.C. § 103(a)

The Examiner has rejected claims 1, 4-7 and 9-10 under 35 U.S.C. § 103(a) as being unpatentable over Akkary et al. ("Akkary," U.S. Patent 6,493,820 B2) in view of Lee et al. ("Lee," U.S. Patent 6,223,276 B1). Applicant respectfully disagrees. The Examiner states:

> a. Regarding claim 1, Akkary et al. discloses a programmable graphics processor (processor 50, Figure 2) for executing a plurality of programs (...thread management logic 124 creates different threads from a program or process... col. 5, lines 24-67; col. 6, lines 1-4), said programmable graphics processor (processor 50) comprising: an execution pipeline (execution pipeline 108) having a pipeline latency (...there may be... various... latency, etc... col. 26, lines 35-40); an interleaver (thread management logic 124) for interleaving instructions (...a thread includes the trace... a trace is a... instruction... col. 5, lines 20-25) from said plurality of programs (...threads are either from completely independent programs or are from the same program... col. 1, lines 63-65) and providing said instructions (...a thread includes the trace... a trace is a... instruction... col. 5, lines 20-25) to said pipeline (execution pipeline 108) for execution. Akkary et al. does not address the issue of each one of said programs having an apparent latency less than said pipeline latency (...there may be... various... latency, etc... col. 26, lines 35-40). Lee et al. discloses the significance of latency problem and addresses this issue in its invention (col. 1, lines 28-44). Therefore, it would have been obvious to one of ordinary skill in the art at the time invention was made to modify the device as taught by Akkary et al. with the feature "programs having an apparent latency less than the said pipeline latency" as taught by Lee et al. because it provides a way to avoid performance bottleneck in a fixed pipeline depth (col. 1, lines 28-44).
>
> b. Regarding claim 4, Akkary et al. discloses wherein each program of said plurality of programs is independent of the other of said plurality of programs (...threads... these processors process and execute are independent of each other... col. 1, lines 58-64).
>
> c. Regarding claim 5, Akkary et al. discloses including an output buffer (ROB 164 and MOB 178) for storing out of order data output (...the result of an execution and related information... written to... re-order buffer (ROB) 164... col. 7, lines 36-50).
>
> d. Regarding claims 6 and 7, Akkary et al. discloses including one or more of a register copy (thread management logic 124), program counter (program counters 112A,... 112X... col. 5, lines 25-30), and program counter stack (thread management logic 124) provided for each of said plurality of programs and further discloses wherein one or more of control and computing resources, instructions, instruction memory, data paths, data memory, and caches are shared by said plurality of programs (Figure 1 and 2).
>
> e. Regarding claims 9 and 10, Akkary et al. discloses wherein said instructions comprise load instructions for loading data from a data memory (load buffers 182, Figure 3), and store instructions

for storing data in a memory (store buffers 184, Figure 3) and wherein said data memory (MOB 178) comprises a cache (data cache 176).

The Examiner has rejected claim 8 under 35 U.S.C. § 103(a) as being unpatentable over Akkary in view of Lee and in further view of Nguyen et al. ("Nguyen," U.S. Patent 5,961,628). Applicant respectfully disagrees. The Examiner states:

> a. Regarding claim 8, Akkary et al. and Lee et al. implicitly discloses SIMD execution of vector instructions without addressing vector lengths. Nguyen et al. explicitly discloses wherein said processor executed SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N (col. 1, lines 11-24; col. 53-60). Therefore, it would have been obvious to one of ordinary skill in the art at the time invention was made to modify the device as taught by Akkary-Lee combination with the feature "SIMD vector instructions execution of vector length L and plurality of instructions having SIMD vector lengths summing up to N" as taught by Nguyen et al. because it provides a way to reduce processing time for repetitive task (col. 1, lines 10-25).

The Examiner has rejected claims 12 and 13 under 35 U.S.C. § 103(a) as being unpatentable over Akkary in view of Lee and in further view of Narayanaswami (U.S. Patent 5,973,705). Applicant respectfully disagrees. The Examiner states:

> a. Regarding claims 12 and 13, Lee et al. does not disclose a graphics processor wherein address space of said data memory comprises a frame buffer unit and a texture memory unit as it describes a vector processor in general with possible suggestion of its use in multimedia processing (col. 1, lines 10-25). Narayanaswami discloses explicitly a SIMD graphics processing system comprising a frame buffer unit (frame buffer 110f, Fig. 2A) while implicitly suggesting a texture memory unit. Therefore, it would have been obvious to one of ordinary skill in the art at the time invention was made to modify the device as taught by Akkary-Lee combination with the feature "frame buffer and texture memory unit" as taught by Narayanaswami because it provides a way to reduce processing time (col. 2, lines 20-22).

The Examiner has rejected claim 2 under 35 U.S.C. § 103(a) as being unpatentable over Akkary in view of Lee and in further view of Dye (U.S. Patent 5,778,250). Applicant respectfully disagrees. The Examiner states:

> a. Regarding claim 2, Akkary-Lee et al. combined teachings does not address the limitation wherein said pipeline (execution pipeline 108) has a datapath with a depth equal to said number of programs. Dye discloses the pipeline stages being dynamically adjusted for simpler or complex operations resulting in increased speed and performance of the processor (col. 3, lines 1-11). Therefore, it would have been obvious to one of ordinary skill in the art at the time invention was made to modify the device as taught by Akkary-Dye combination with the feature "dynamically adjustable pipeline stages of an execution pipeline" as taught by Dye because it provides a way to improve the overall speed and performance of the processor (col. 3, lines 1-11; col. 5, lines 5-10).

The Examiner has rejected claims 3 and 14-18 under 35 U.S.C. § 103(a) as being unpatentable over Akkary in view of Lee and in further view of Naini et al. ("Naini," U.S. Patent 6,209,083). Applicant respectfully disagrees. The Examiner states:

> b. Regarding claim 3, Lee et al. does not disclose wherein a next instruction from one of said plurality of programs (...threads are either from completely independent programs or are from the same program... col. 1, lines 63-65) is not provided to said pipeline (execution pipeline 108) until a previous instruction of said one of said plurality of programs (...threads are either from completely independent programs or are from the same program... col. 1, lines 63-65) has completed. Naini et al. discloses working in the same respect at the claim limitation "...processor will not issue a next... instruction... until the previously issued... instruction has cleared... col. 2, lines 1-5". Naini et al. further indicates that the previous instruction will not have an exception (col. 2, lines 1-5). The application specification is clear in detailing avoiding the hardware complexity of pipeline bypasses, instruction reordering or the inefficiencies of idle cycles (page 11, 1st paragraph) in much the same fashion. Therefore, it would have been obvious to one of ordinary skill in the art at the time invention was made to modify the device as taught by Akkary-Lee combination with the feature "no next instruction into the pipeline until the previous instruction has completed or retired from the pipeline" as taught by Naini et al. because it provides a way to reduce pipeline stalling, or the need for pipeline bypass, instruction reordering or idle cycles in the pipeline (col. 1, lines 59-60).
>
> c. Regarding claim 14, it is similar in scope to claim 3 above and is rejected under the same rationale.
>
> d. Regarding claims 15 and 16, they are similar in scope to claim 6 above and are rejected under the same rationale.
>
> e. Regarding claim 17, it is similar in scope to claim 2 above and is rejected under the same rationale.
>
> f. Regarding claim 18, it is similar in scope to claim 8 above and is rejected under the same rationale.

Applicant respectfully disagrees. Applicant submits that claims 1-18 are not allowable over Akkary in view of Lee and in further view of Nguyen, Narayanaswami, Dye and/or Naini for at least the following reasons.

1. The present invention is not unpatentable over Akkary in view of Lee and in further view of Nguyen, Narayanaswami, Dye and/or Naini because they do not teach an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline.

Examiner relies on Akkary in view of Lee and in further view of Dye to teach an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that

are interleaved is greater than or equal to the depth of the pipeline. However, none of Akkary, Lee or Dye teaches an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline. Akkary teaches multithreading processors executing multiple threads concurrently, but it does teach, describe or suggest that the number of programs to be interleaved is greater than or equal to the depth of the processor pipeline.

Examiner seems to rely mostly upon Dye to teach the number of programs being equal to the depth of the processor pipeline; however, Dye does not teach any relation between the processor pipeline depth and a number of programs interleaved for execution. Instead, Dye teaches only a dynamic pipeline depth without any mention of a number of programs to be interleaved for execution.

For at least the forgoing reasons, Applicant submits that the cited reference does not teach, describe or suggest the present invention and that independent claim 1 and all dependent claims are patentably distinct from the cited prior art.

2. The present invention is not unpatentable over Akkary in view of Lee and in further view of Nguyen, Narayanaswami, Dye and/or Naini because they do not teach identifying N programs of said plurality of programs wherein N is greater than or equal to the depth of a processor pipeline.

Examiner relies on Akkary in view of Lee and in further view of Dye to teach identifying N programs of said plurality of programs wherein N is greater than or equal to the depth of a processor pipeline. However, none of Akkary, Lee or Dye teaches identifying N programs of said plurality of programs wherein N is greater than or equal to the depth of a processor pipeline. Akkary teaches multithreading processors executing multiple threads concurrently, but it does

teach, describe or suggest that the number of programs to be interleaved, N, is greater than or equal to the depth of the processor pipeline.

Examiner seems to rely mostly upon Dye to teach the number of programs being equal to the depth of the processor pipeline; however, Dye does not teach any relation between the processor pipeline depth and a number of programs interleaved for execution. Instead, Dye teaches only a dynamic pipeline depth without any mention of a number of programs to be interleaved for execution.

For at least the forgoing reasons, Applicant submits that the cited reference does not teach, describe or suggest the present invention and that independent claim 23 and all dependent claims are patentably distinct from the cited prior art.

3. The present invention is not unpatentable over Akkary in view of Lee and in further view of Nguyen, Narayanaswami, Dye and/or Naini because they do not teach an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed.

Examiner relies on Akkary in view of Lee and in further view of Naini to teach an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed. However, none of Akkary, Lee or Naini teaches an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for

execution wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed. Akkary teaches multithreading processors executing multiple threads concurrently, but it does teach, describe or suggest that execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline without the addition of no-ops to ensure that result.

Examiner seems to rely mostly upon Naini to teach that execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline; however, Naini does not teach execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline without the addition of no-ops to ensure that result. Instead, Naini teaches inserting stalls (i.e., no-ops) in the pipeline to ensure that execution of one instruction of a program is completed before a subsequent instruction from the same program enters the pipeline.

For at least the forgoing reasons, Applicant submits that the cited reference does not teach, describe or suggest the present invention and that independent claim 24 and all dependent claims are patentably distinct from the cited prior art.

4. The present invention is not unpatentable over Akkary in view of Lee and in further view of Nguyen, Narayanaswami, Dye and/or Naini because they do not teach executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of said one of said N programs wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction.

Examiner relies on Akkary in view of Lee and in further view of Naini to teach executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of said one of said N programs wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction. However, none of Akkary, Lee or Naini teaches executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of said one of said N programs wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction. Akkary teaches multithreading processors executing multiple threads concurrently, but it does teach, describe or suggest that execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline without the addition of no-ops to ensure that result.

Examiner seems to rely mostly upon Naini to teach that execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline; however, Naini does not teach execution of an instruction of one program is completed before a subsequent instruction from the same program enters the pipeline without the addition of no-ops to ensure that result. Instead, Naini teaches inserting stalls (i.e., no-ops) in the pipeline to ensure that execution of one instruction of a program is completed before a subsequent instruction from the same program enters the pipeline.

For at least the forgoing reasons, Applicant submits that the cited reference does not teach, describe or suggest the present invention and that independent claim 14 and all dependent claims are patentably distinct from the cited prior art.

For at least the foregoing reasons, Applicant submits that the cited reference does not teach, describe or suggest the present invention. Therefore, Applicant submits that independent claims 1, 14 and 23-24 are allowable. Further, dependent claims 2-18, being dependent upon respective allowable base claims, are also allowable for at least the forgoing reasons with respect to claims 1, 14 and 23-24.

CONCLUSION

Attached hereto is a marked-up version of the changes made to the claims by the current amendment. The attached page is captioned **"Version with markings to show changes made."**

For at least the foregoing reasons, Applicant respectfully submits that pending claims 1-18, 23 and 24 are patentably distinct from the prior art of record and in condition for allowance. Applicant therefore respectfully requests that pending claims 1-18, 23 and 24 be allowed.

Respectfully submitted,

COUDERT BROTHERS LLP

Date: May 12, 2003

By: _____
J. D. Harriman
Reg. No. 31,967

VERSION WITH MARKINGS TO SHOW CHANGES MADE

1. A programmable [graphics] processor for executing a plurality of programs, said programmable [graphics] processor comprising:

an execution pipeline having a pipeline latency; and

an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that [each one of said programs has an apparent latency less than said pipeline latency] the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline.

2. The [graphics] processor of claim 1 or claim 23 wherein said pipeline has a datapath with a depth equal to said number of programs.

3. The [graphics] processor of claim 1 wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed.

4. The [graphics] processor of claim 1 or claim 23 wherein each program of said plurality of programs is independent of the other of said plurality of programs.

5. The [graphics] processor of claim 1 or claim 23 further including an output buffer for storing out of order data output.

6. The [graphics] processor of claim 1 or claim 23 further including one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs.

7. The [graphics] processor of claim 6 wherein one or more of control and computing resources, instructions, instruction memory, data paths, data memory, and caches are shared by said plurality of programs.

8. The [graphics] processor of claim 1 or claim 23 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.

9. The [graphics] processor of claim 1 or claim 23 wherein said instructions comprise load instructions for loading data from a data memory.

10. The [graphics] processor of claim 1 or claim 23 wherein said instructions comprise store instructions for storing data in a memory.

11. The [graphics] processor of claim 9 wherein said data memory comprises a cache.

12. The [graphics] processor of claim 9 wherein address space of said data memory comprises a frame buffer unit.

13. The [graphics] processor of claim 9 wherein address space of said data memory comprises a texture memory unit.

14. A method of executing instructions from a plurality of [graphic processing] programs comprising:

identifying N programs of said plurality of programs;

interleaving instructions from said N programs in a [graphics processing execution] processor pipeline; and

executing said instructions such that a first instruction from [any] one of said N programs is completed before beginning execution of a second instruction of [any] said one of said N programs wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction.

15. The method of claim 14 or claim 24 further including the step of assigning a program counter to each of said N programs.

16. The method of claim 14 or claim 24 further including the step of assigning a register to each of said N programs.

17. The method of claim 14 or claim 24 wherein said graphics processing execution pipeline has a depth of N.

18. The method of claim 14 or claim 24 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.